数据结构课程设计

题目八 电网建设造价模拟系统

文档

班级：软件5班

学号：1353010

姓名：薛梦迪

**一、项目简介：**

假设一个城市有n个小区，要实现n个小区之间的电网都能够相互接通，构造这个城市n个小区之间的电网，使总工程造价最低。请设计一个能够满足要求的造价方案。

**二、项目功能：**

在每个小区之间都可以设置一条电网线路，都要付出相应的经济代价。n个小区之间最多可以有n（n-1）/2条线路，选择其中的n-1条使总的耗费最少。

在数据结构方面讲，即建立一个有向图，并用prim算法生成其最小生成树，这个最小生成树即可以使总店耗费最少。

具体功能：

1、创建电网顶点：选择操作A，并输入顶点个数和各顶点名称。

2、添加电网的边：选择操作B，输入两个顶点及边，输入“0 0 0”作为退出符，若边建立失败则会返回“Failed to insert edge of XX and XX of cost of XX”。

3、构造最小生成树：选择操作C，输入起始顶点，建立最小生成树。

4、显示最小生成树：选择操作D，自动显示最小生成树。

5、退出：选择操作E，退出系统。

[注：操作不分大小写]

**三、项目数据格式说明与类说明：**

1、PQueue.h

1）引用的头文件：

#include"assert.h"

#include"stdlib.h"

2）内容：最小优先级队列 PQueue 类定义

template <class T>

|  |  |  |  |
| --- | --- | --- | --- |
| **访问控制类型** | **成员名称** | **成员类型** | **成员功能** |
| public | Pqueue(PQueue(int sz = DefaultPQSize) | constructor | 构造函数，构造一个最小优先级队列 |
| Insert(const T& x) | bool | 将新元素x插入到队尾 |
| RemoveMin(T& x) | bool | 将队头元素删去 |
| getFront(T& x)const | bool | 读取队头（具最小优先权）的值 |
| makeEmpty() | void | 置优先级队列为空 |
| IsEmpty()const | bool | 判队列空否 |
| IsFull()const | bool | 判队列满否 |
| getSize()const | int | 求优先级队列中的元素个数 |
| protected | pqelements | T\* | 优先级队列数组 |
| count | int | 当前元素个数（长度） |
| maxSize | int | 队列最大可容纳元素个数 |
| adjust() | void | 队列调整 |

2、heap.h

1）引用的头文件：

#include"PQueue.h"

#include<iostream>

2）内容：最小堆MinHeap 类定义（继承自PQueue类）

template <class E>

|  |  |  |  |
| --- | --- | --- | --- |
| **访问控制类型** | **成员名称** | **成员类型** | **成员功能** |
| public | MinHeap(int sz = DefaultSize) | constructor | 构造函数：建立空堆 |
| MinHeap(E arr[], int n) | constructor | 构造函数：通过一个数组建堆 |
| Insert(const E& x) | bool | 将x插入到最小堆中 |
| RemoveMin(E& x) | bool | 删除堆顶上的最小元素 |
| IsEmpty()const | bool | 判断空否，空返回1，否则0 |
| IsFull()const | bool | 判断满否，满返回1，否则0 |
| MakeEmpty() | void | 置空堆 |
| private | heap | E \* | 存放最小堆中元素的数组 |
| currentSize | int | 最小堆中当前元素个数 |
| maxHeapSize | int | 最小堆最多允许元素个数 |
| siftDown(int start,int m) | void | 从start到m下滑调整成为最小堆 |
| siftUp(int start) | void | 上滑调整最小堆 |

3、GraphInk.h

1）引用的头文件：

#include"heap.h"

2）内容：

i）图的边结构体 Edge

template <class T>

|  |  |  |
| --- | --- | --- |
| **成员名称** | **成员类型** | **成员功能** |
| dest | int | 边的另一顶点位置 |
| cost | int | 边上的权值 |
| link | Edge<T> \* | 下一条边链指针 |
| Edge() | constructor | 构造函数 |
| Edge(int num,int weight) | constructor | 构造函数 |
| operator != | bool | 判断边是否相等 |

ii）图的结点结构体 Vertex

template <class T>

|  |  |  |
| --- | --- | --- |
| **成员名称** | **成员类型** | **成员功能** |
| data | T | 顶点的名字 |
| adj | Edge<T> \* | 边链表的头指针 |

iii）图的邻接表表示GraphInk 类定义

template <class T>

|  |  |  |  |
| --- | --- | --- | --- |
| **访问控制类型** | **成员名称** | **成员类型** | **成员功能** |
| public | GraphInk(int sz = 30) | constructor | 构造函数：建立一个空的图 |
| getValue(int i) | T | 取位置为i的顶点中的值 |
| getWeight(int v1,int v2) | int | 返回边(v1,v2)上的权值 |
| insertVertex(const T& vertex) | bool | 在图中插入一个顶点vertex |
| insertEdge(int v1,int v2,int weight) | bool | 在图中插入一条边(v1,v2) |
| getFirstNeighbor(int v) | int | 取顶点v的第一个邻接顶点 |
| getNextNeighbor(int v,int w) | int | 取v的邻接顶点w的下一邻接顶点 |
| findVerInsertEdge(T v1,T v2,int vcost) | bool | 按照名称寻找(v1,v2)边并插入此条边，其中权值为vcost |
| NumberOfVertices() | int | 返回顶点数量 |
| NumberOfEdges() | int | 返回边的数量 |
| DFS(GraphInk<T>& G,int v,bool visited[]) | void | 遍历打印子过程 |
| DFS(GraphInk<T>& G,const T& v) | void | 遍历打印主过程 |
| getVertexPos(const T vertex) | int | 给出顶点vertex在图中的位置 |
| private | maxVertices | int | 图中最大顶点数 |
| numEdges | int | 当前边数 |
| numVertices | int | 当前顶点数 |
| NodeTable | Vertex<T> \* | 顶点表 |

4、MST.h

1）引用的头文件：

#include"float.h" #include"GraphInk.h"

2）内容：

i）最小生成树边结构体MSTEdgeNode

|  |  |  |
| --- | --- | --- |
| **成员名称** | **成员类型** | **成员功能** |
| tail | int | 尾顶点位置 |
| head | int | 头顶点位置 |
| key | int | 两顶点之间的边上的权值 |
| MSTEdgeNode() | constructor | 构造函数 |
| operator <= | bool | 重载运算符 |
| operator > | bool | 重载运算符 |

ii）最小生成树MinSpanTree类定义

template <class T>

|  |  |  |  |
| --- | --- | --- | --- |
| **访问控制类型** | **成员名称** | **成员类型** | **成员功能** |
| public | MinSpanTree(int sz = 30) | constructor | 构造函数 |
| Insert(MSTEdgeNode& item) | int | 插入最小生成树的一个边结点 |
| print(GraphInk<T>& G) | void | 打印最小生成树 |
| protected | edgevalue | MSTEdgeNode \* | 用边值数组表示树 |
| MaxSize | int | 数组的最大元素个数 |
| n | int | 数组当前元素个数 |

iii）最小生成树Prim算法

template <class T>

void Prim(GraphInk<T>& G,const T u0,MinSpanTree<T>& MST){

MSTEdgeNode ed;int v,count;

int n = G.NumberOfVertices();

int m = G.NumberOfEdges();

int u = G.getVertexPos(u0);

MinHeap<MSTEdgeNode> H(m);

bool Vmst[n];

for(int i=0;i<n;i++)

Vmst[i] = false;

Vmst[u] = true;

count = 1;

do{

v = G.getFirstNeighbor(u);

while(v!=-1){

if(Vmst[v]==false){

ed.tail = u;

ed.head = v;

ed.key = G.getWeight(u,v);

H.Insert(ed);

}

v = G.getNextNeighbor(u,v);

}

while(H.IsEmpty() == false && count < n){

H.RemoveMin(ed);

if(Vmst[ed.head] == false){

MST.Insert(ed);

u = ed.head;

Vmst[u] = true;

count ++;

break;

}

}

} while(count < n);

}

**四、项目运行的界面：**



注：1、完成了基本操作。

2、边建立失败会返回错误信息。

3、选择不分大小写。

**五、项目优点和不足之处：**

本项目满足了项目的所有需求，并在输入正确的情况下能顺利执行，并且支持中文、英文字母和单词作为数据输入、输出。本项目也基本遵循了类的封装性，较好的保护了数据，更好的保存了数据。由于类非常多，没有过多时间加入注释，这是项目的疏漏之处。总之，本项目具有较强的客观性，也难免有一些小的地方考虑不周，希望以后能更加改进。